

COC2626 Cloud computing 2021 - Assignment 2
RMIT University

Victoria Road Accident Analytics

Nithi Rakkisiri (s376900)
Jakrapun Sangchan (s3808216)

Contribution Agreement	3
Links	4
Summary	4
Introduction	4
Motivation	4
How does it work?	4
How can it be used as a real-life application?	4
Related work	5
Software Design/Architecture	7
A high-level architecture diagram	7
Description of your data set.	8
Implementation	9
Setup Elastic Beanstalk Environment for web server	9
Create AWS S3 Bucket	10
Create structured table in S3 Bucket using Amazon Athena	12
Create Dashboard using Amazon Quicksight	13
Connect with Google Map API	15
Create Aws Lambda function	15
Prepare Nodejs Lambda project on local machine	16
Create API Gateway	17
Install SSL certificate on Elastic Beanstalk Load Balancer using ACM and Route53	20
User manual	24
Road Accident Analysis	24
Interactive Map	25
Export Report	26
Reference.	26


Contribution Agreement



RMIT Classification: Trusted

Appendix: Student Contribution Agreement

Victoria Road Accident Analytics

Student Name: Jakrapun Sangchan	Student Name: Nithit Rakkisiri
Student ID: s3808216	Student ID: s376900
Contributions: 1. QuickSight dashboard design 2. Create athena table 3. Handle IAM policy for QuickSight	Contributions: 1. Create Lambda function and API Gateway 2. Connect google map API 3. Deploy webApp and API to Elastic Beanstalk
Contribution Percentage: 50	Contribution Percentage: 50
By signing below, I certify all information is true and correct to the best of my knowledge.	By signing below, I certify all information is true and correct to the best of my knowledge.
Signature: 	Signature: 
Date: 30/01/2021	Date: 30/01/2021

Links

<https://www.cloud-project-js-nr.com/>

Summary

This project focuses on analysing the road accident data across Victoria and create a user-friendly interface to understand the analysis result. We use a public dataset from Vicroads data that record road accidents across Victoria from July 2013 to March 2019. We decided to use AWS quick sight to get insight from the data and create an online dashboard. Google map API has been used to illustrate the location and detail of accidents.

Introduction

Motivation

The motivation of this project is to create a simplified dashboard about characteristics of road accidents in Victoria to help stakeholders understand which factor contributes to the accident.

How does it work?

Our Analytics app has three parts.

The first feature is an **analytic dashboard** that shows summary statistics of road accidents between Jul 2013 and March 2019. The dashboard shows simplified charts and a short summary of each chart. The Dashboard section has a print feature that users can export the dashboard as a PDF file.

The second part **maps**, which shows the location of recorded accidents on the map according to user filter preference.

The third part is the **export data**, this feature lets the user can export the full dataset of this analytics.

How can it be used as a real-life application?

This project demonstrates how to create data analytics on cloud environments. This project would like to simplify cloud data analytics web application. So, people could use this project framework to implement another data analytics web application.

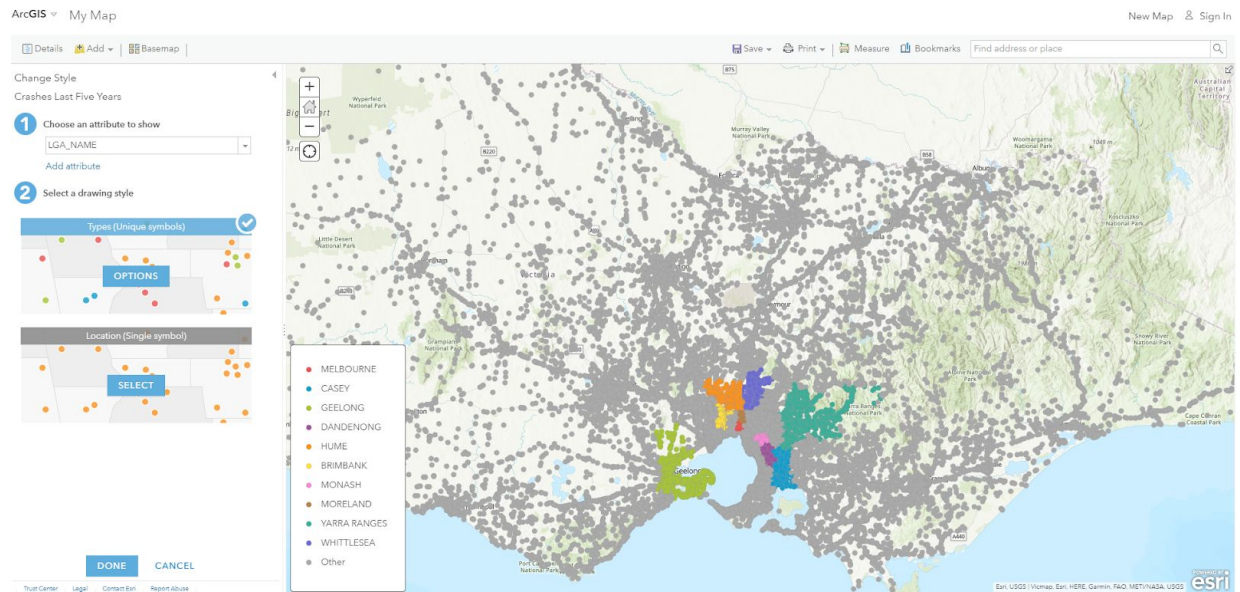
Related work

There are many works that are related to this project. The visualization below is one of the examples of road accident analytics from Spatial Vision company. The visualization shows statistical information about road accidents. [2]



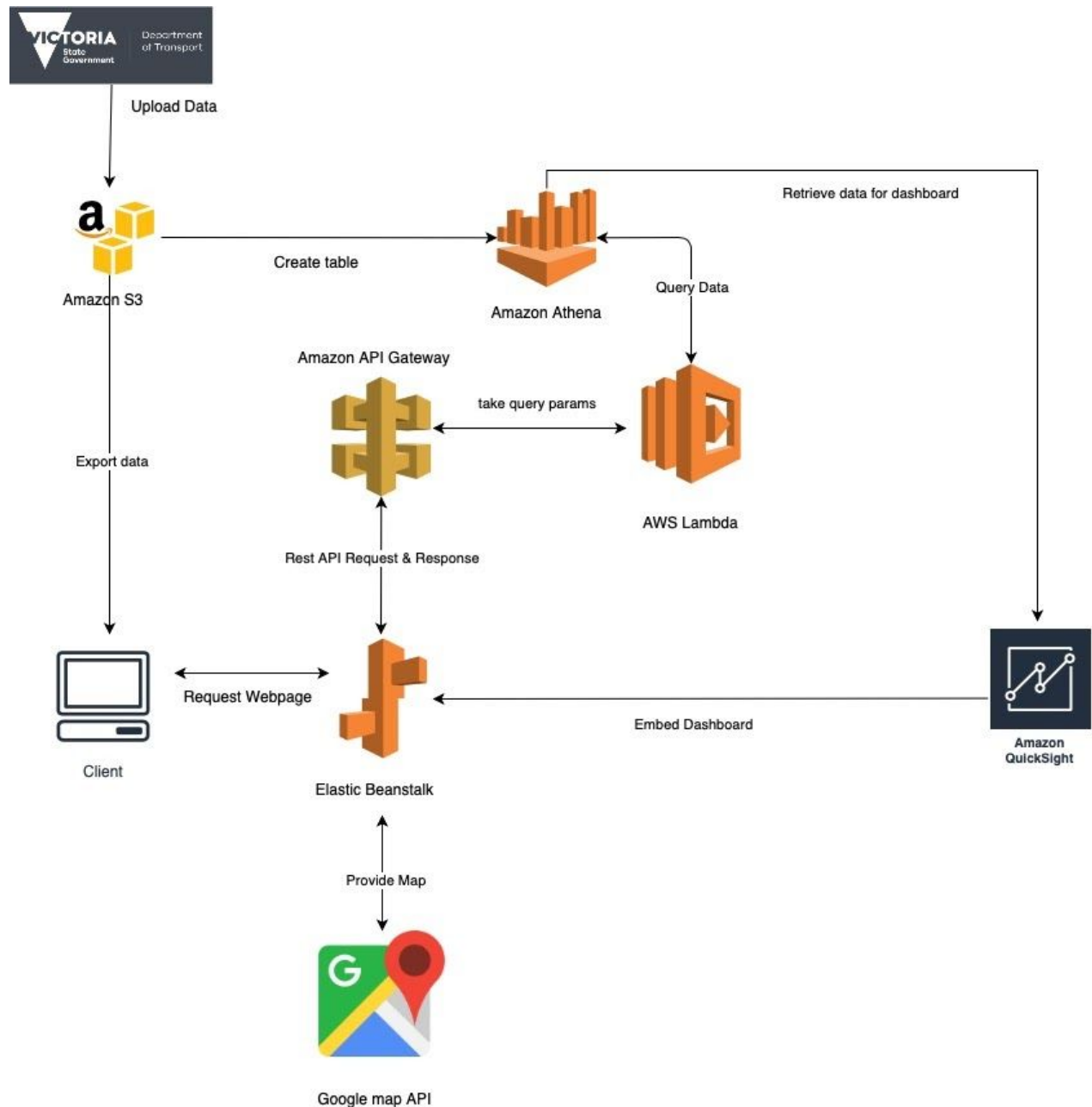
<https://www.spatialvision.com.au/crashstats/>

Other related work is on Vicroads website. The visualization shows the location of an accident with one attribute such as LGA_NAME. [1]



https://www.arcgis.com/home/webmap/viewer.html?panel=gallery&suggestField=true&url=https%3A%2F%2Fservices2.arcgis.com%2F18ajPSI0b3ppsmMt%2Farcgis%2Frest%2Fservices%2FCrashes_Last_Five_Years%2FFeatureServer%2F0

Software Design/Architecture



A high-level architecture diagram

- Services and framework
 - Services
 - Elastic Beanstalk
 - We use elastic beanstalk to host the web server with Node.js 12 running on 64bit Amazon Linux 2/5.2.4. Because of using embedded dashboard from Aws Quicksight require valid certificates for the web server, so

additional configuration is required eg. add listener port (443) for https and public domain is used.

- Aws API Gateway
 - We use Rest API protocol to handle all requests from our web server.
- Aws Lambda
 - We use Lambda proxy integration with Aws API Gateway. The Lambda function using Nodejs runtime environment which will handle data queries from web servers and retrieve data from S3 by Aws Athena. This Lambda function also generates signed url for S3 export file that allow users to download the report.
- Amazon Athena
- Amazon Quicksight
- Amazon S3
- Google Map API
 - We use Google Map API to generate map and pin the accident location from queries with detail in each point.
- Amazon Code Commit
 - We've decided to use code commit to deploy code to ElasticBeanstalk from git so, we can easily manage the code version and manage deployment version and also collaborate between teammates.
- Framework
 - Nodejs
 - We use express framework of Nodejs for API part in aws Lambda function
 - We use ejs template for frontend part

Description of your data set.

We use Crashes Last Five Years dataset from Vicroads. We use Amazon S3 to store the dataset in CSV format and Amazon Athena to query the dataset. The dataset has 74,908 records of road accidents since July 2013 to Feb 2019. The dataset consists of many types of information such as road geometry, light condition, location (latitude and longitude) and severity.

Showing 1 to 10 of 74,908



Hint: Filter columns using 

ACCIDENT_NO	ABS_CODE	ACCIDENT_STATUS	ACCIDENT_DATE	ACCIDENT_TIME	ALCOHOLTIME	ACCIDENT_TYPE
T20130013732	ABS to receive accident	Finished	1/7/2013	18.30.00	Yes	Struck Pedestrian
T20130013736	ABS to receive accident	Finished	2/7/2013	16.40.00	No	Collision with vehicle
T20130013737	ABS to receive accident	Finished	2/7/2013	13.15.00	No	Collision with a fixed object
T20130013738	ABS to receive accident	Finished	2/7/2013	16.45.00	No	Collision with a fixed object
T20130013739	ABS to receive accident	Finished	2/7/2013	15.48.00	No	Collision with vehicle
T20130013740	ABS to receive accident	Finished	2/7/2013	13.40.00	No	Collision with a fixed object
T20130013742	ABS to receive accident	Finished	2/7/2013	16.10.00	No	Collision with vehicle
T20130013743	ABS to receive accident	Finished	2/7/2013	17.50.00	No	Collision with vehicle
T20130013744	ABS to receive accident	Finished	2/7/2013	17.30.00	No	Collision with vehicle
T20130013746	ABS to receive accident	Finished	2/7/2013	17.35.00	No	Collision with vehicle

Implementation

1. Setup Elastic Beanstalk Environment for web server

1.1. Setup Aws ElasticBeanstalk using a web server environment with

All environments   Create a new environment									
<input type="text" value="Filter results matching the display values"/>									
	Environment name ▲	Health ▼	Application name ▼	Date created ▼	Last modified ▼	URL	Running versions ▼	Platform ▼	Tier name ▼
<input type="radio"/>	node-express-dev	Ok	node-express	2021-01-19 17:43:38 UTC+1100	2021-01-24 23:46:56 UTC+1100	node-express-dev22.us-east-1.elasticbeanstalk.com	app-1a26-210124_234612	Node.js 12 running on 64bit Amazon Linux 2	WebServer

Nodejs 12 runtime. Aws will create an ec2 instance , load balancer , cloud watch and nginx proxy server for this environment.

1.2. Since we decided to use Codecommit for deploying the code to Elastic Beanstalk this operation will need to Install Aws cli and Elastic Beanstalk cli on local machines.

1.2.1. On root project directory init git and elastic beanstalk using command :

```
$ git init
$ eb init
```

1.2.2. Next step is to check out master branch and configure the Elastic Beanstalk environment

```
$ git checkout master
$ eb use <env name>
```

1.2.3. Add changes to git and commit change

\$ git add .

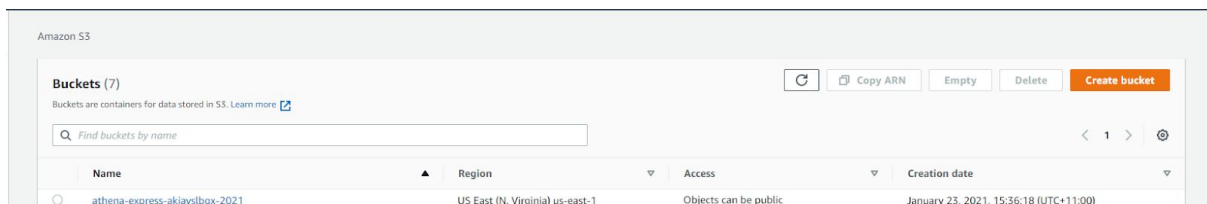
\$ git commit

1.2.4. Deploy committed code to ElasticBeanstalk environment

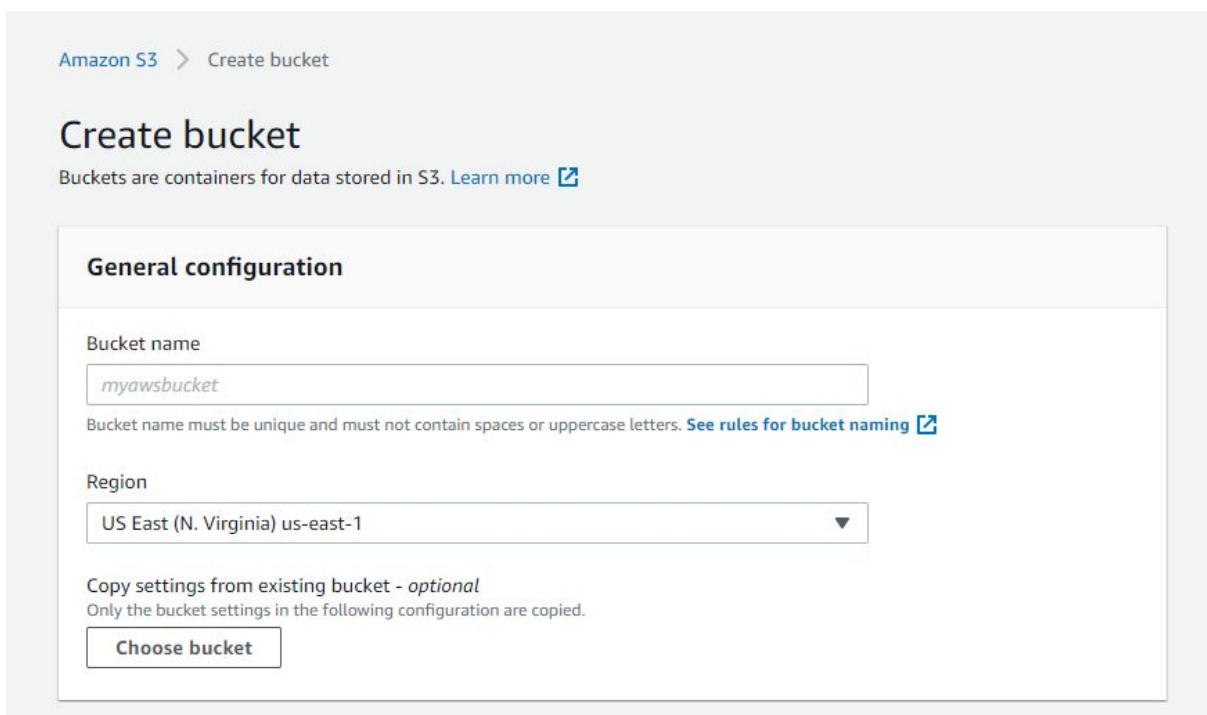
\$ eb deploy

2. Create AWS S3 Bucket

2.1. Click create bucket at Amazon s3



2.2. Name your bucket and create a bucket.



2.3. Upload CSV file to the bucket by using the Add files button.

[Amazon S3](#) > [crashes-data1](#) > Upload

Upload

Add the files and folders you want to upload to S3. To upload a file larger than 160GB, use the AWS CLI, AWS SDK or Amazon S3 REST API. [Learn more](#)

Drag and drop files and folders you want to upload here, or choose **Add files**, or **Add folders**.

Files and folders (0)

Remove

Add files

Add folder

All files and folders in this table will be uploaded.

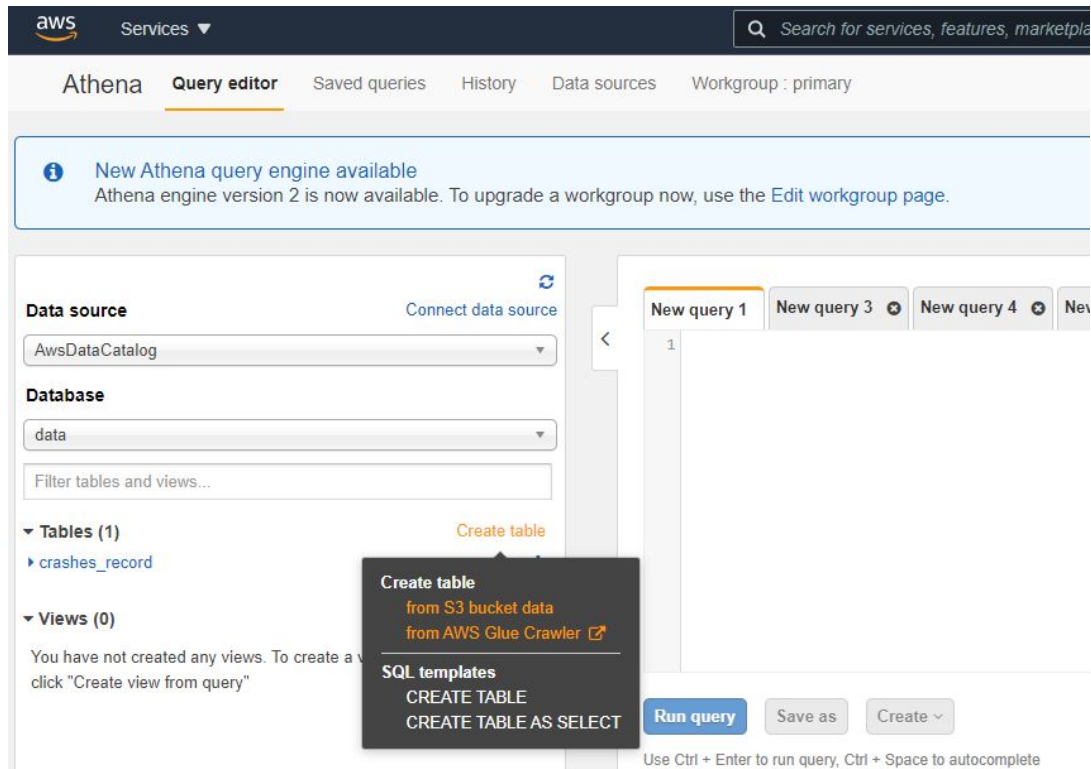
Find by name

< 1 >

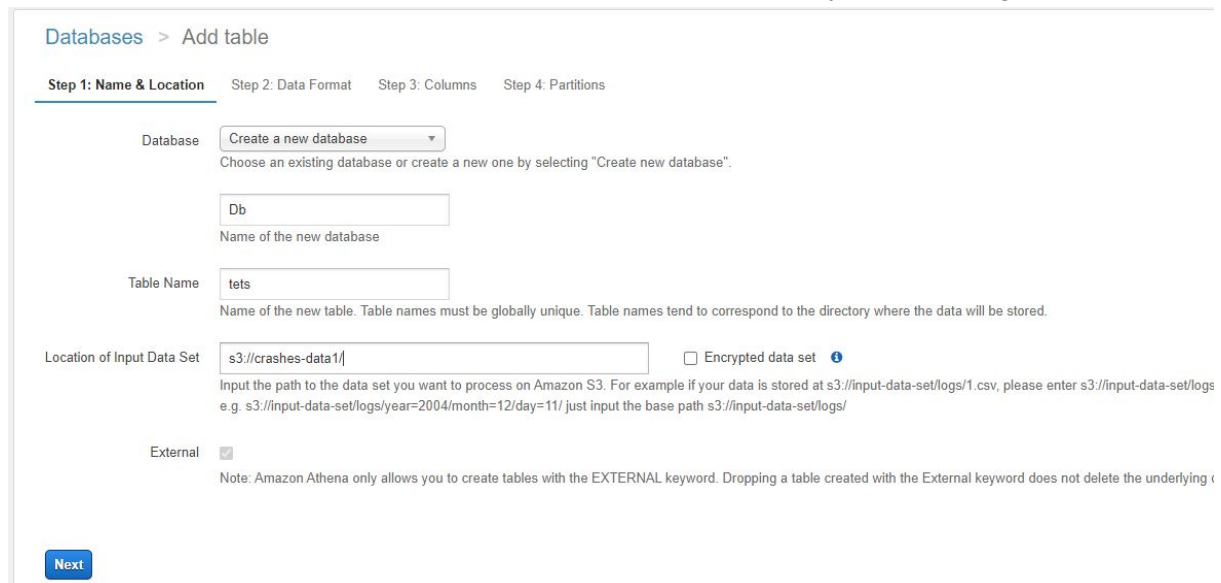
	Name	Folder	Type	Size
No files or folders				
You have not chosen any files or folders to upload.				

3. Create structured table in S3 Bucket using Amazon Athena

3.1. Go to Athena console and select Create table from S3 bucket data.



3.2. Add database name, table name and location of your s3 storage



3.3. Select Bulk add columns and add the following script below.

objectid int, accident_no string, abs_code string, accident_status string, accident_date string, accident_time string, alcoholtime string, accident_type string, day_of_week string, dca_code string, hit_run_flag string, light_condition string, police_attend string, road_geometry string, severity string, speed_zone string, run_offroad string, node_id int, longitude double, latitude double, node_type string, lga_name string, region_name string, vicgrid_x int, vicgrid_y int, total_persons int, inj_or_fatal int, fatality int, seriousinjury int, otherinjury int, noninjured int, males int, females int, bicyclist int, passenger int, driver int, pedestrian int, pillion int, motorist int, unknown int, ped_cyclist_5_12 int, ped_cyclist_13_18 int, old_pedestrian int, old_driver int, young_driver int, alcohol_related string, unlicencsed int, no_of_vehicles int, heavyvehicle int, passengervehicle int, motorcycle int, publicvehicle int, deg_urban_name string, deg_urban_all string, lga_name_all string, region_name_all string, srns string, srns_all string, rma string, rma_all string, divided string, divided_all string, stat_div_name string

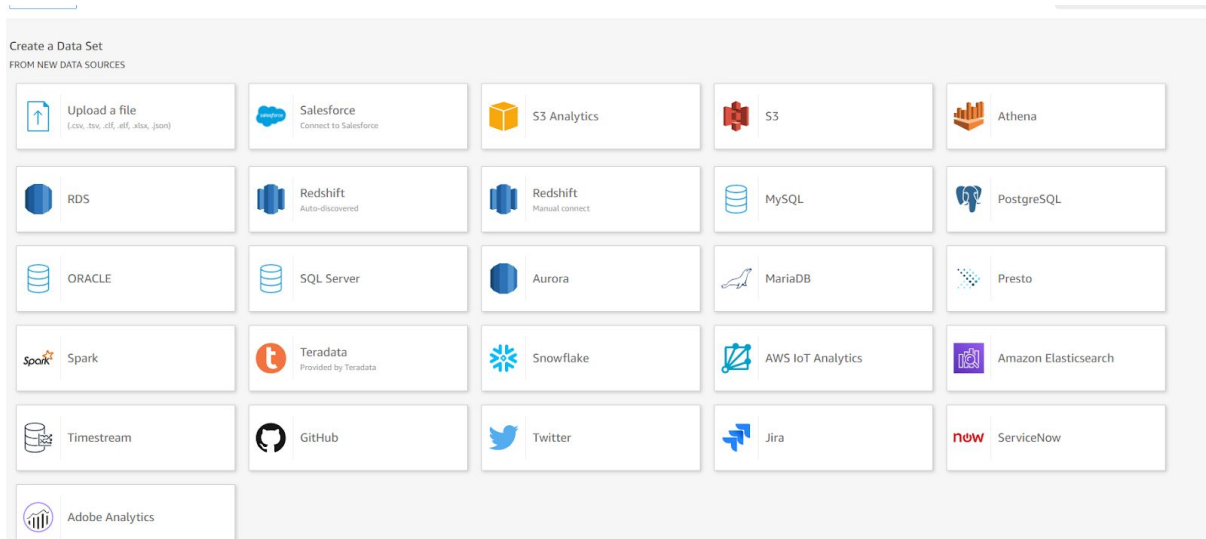
3.4. Select Create table.

4. Create Dashboard using Amazon Quicksight

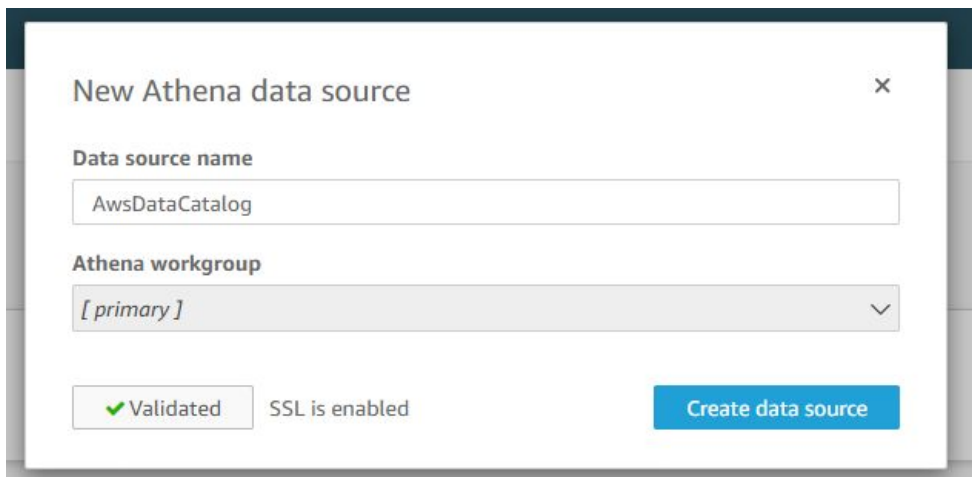
4.1. Go to aws QuickSight select Datasets on the left panel

4.2. New dataset

4.3. Select Athena Data source

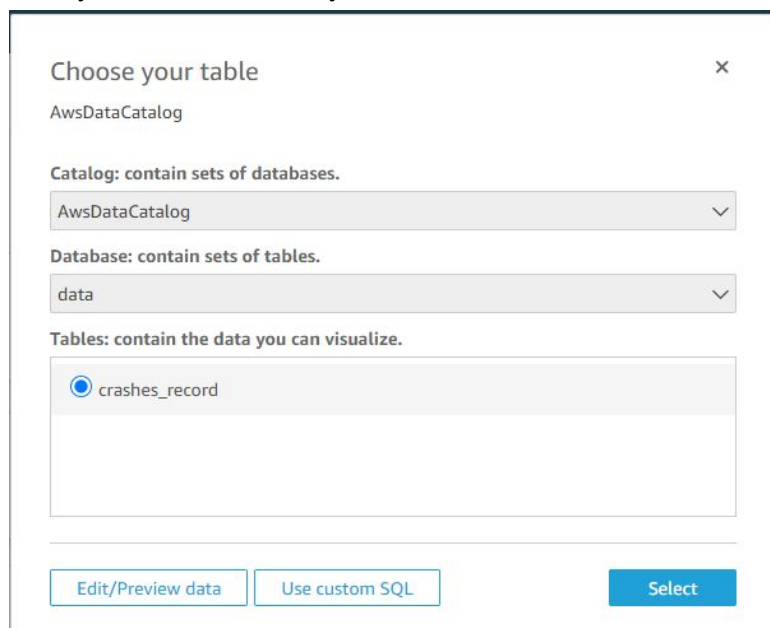


- 4.4. Enter your data source name and validate connection If connection has been validated create data source



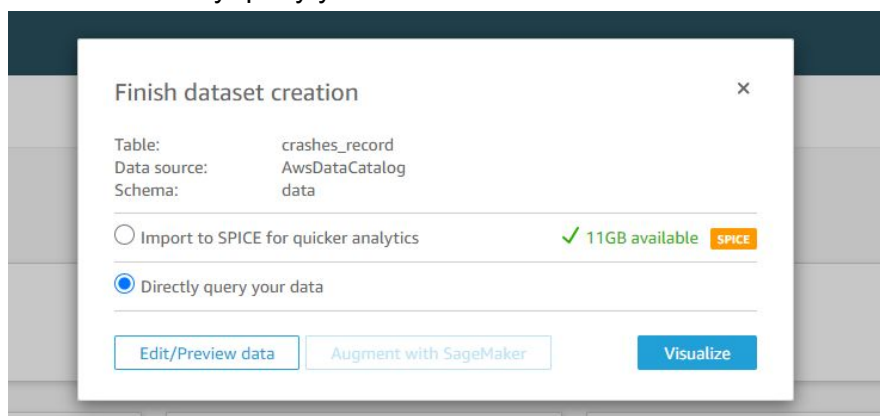
A dialog box titled "New Athena data source" with a close button (X) in the top right corner. It contains two input fields: "Data source name" with the text "AwsDataCatalog" and "Athena workgroup" with a dropdown menu showing "[primary]". Below these fields, there is a green checkmark icon followed by the text "Validated" and "SSL is enabled". At the bottom right, there is a blue button labeled "Create data source".

- 4.5. Select your database and your table



A dialog box titled "Choose your table" with a close button (X) in the top right corner. It shows the selected data source "AwsDataCatalog". Below this, there are three sections: "Catalog: contain sets of databases." with a dropdown menu showing "AwsDataCatalog", "Database: contain sets of tables." with a dropdown menu showing "data", and "Tables: contain the data you can visualize." with a list of tables. The table "crashes_record" is selected with a radio button. At the bottom, there are three buttons: "Edit/Preview data", "Use custom SQL", and "Select".

- 4.6. Select Directly query your data and Visualize



A dialog box titled "Finish dataset creation" with a close button (X) in the top right corner. It displays the selected table "crashes_record", data source "AwsDataCatalog", and schema "data". Below this, there are two radio button options: "Import to SPICE for quicker analytics" (which is disabled and has a green checkmark and "11GB available" next to it) and "Directly query your data" (which is selected). At the bottom, there are three buttons: "Edit/Preview data", "Augment with SageMaker", and "Visualize".

- 4.7. After that create charts using tools on the left panel and create text boxes with insights function.
- 4.8. Share your analysis by publishing a dashboard.

5. Connect with Google Map API

- 5.1. Initialize google map API using inline loading by adding inline javascript on html template this step required API key from google cloud platform.

```
<script defer
src="https://maps.googleapis.com/maps/api/js?key=
API_KEY&callback=initMap">
</script>
```

- 5.2. Now we can use map function in our javascript code.

6. Create Aws Lambda function

- 6.1. Create Aws Lambda function by aws console and click on create function button

Functions (3)						
<div> <div> <div></div> <div>Filter by tags and attributes or search by keyword</div> </div> <div> <div>Last fetched now</div> <div> <div></div> <div>Actions</div> </div> <div>Create function</div> </div> </div>						
	Function name	Description	Package type	Runtime	Code size	Last modified
<input type="radio"/>	GetStartedLambdaProxyIntegration		Zip	Node.js 12.x	772 bytes	2 days ago
<input type="radio"/>	GetStartedLambdaIntegration		Zip	Node.js 12.x	562 bytes	2 days ago
<input type="radio"/>	roadAccident	roadAccident	Zip	Node.js 12.x	2.1 MB	yesterday

- 6.2. Choose Author from scratch, fill in project name and select Nodejs runtime.

Create function

Info

Choose one of the following options to create your function.

Author from scratch

Start with a simple Hello World example.

Use a blueprint

Build a Lambda application from sample code and configuration presets for common use cases.

Container image

Select a container image to deploy for your function.

Browse serverless app repository

Deploy a sample Lambda application from the AWS Serverless Application Repository.

Basic information

Function name

Enter a name that describes the purpose of your function.

myFunctionName

Use only letters, numbers, hyphens, or underscores with no spaces.

Runtime

Info

Choose the language to use to write your function.

Node.js 12.x

Permissions

Info

By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.

Change default execution role

Execution role

Choose a role that defines the permissions of your function. To create a custom role, go to the IAM console.

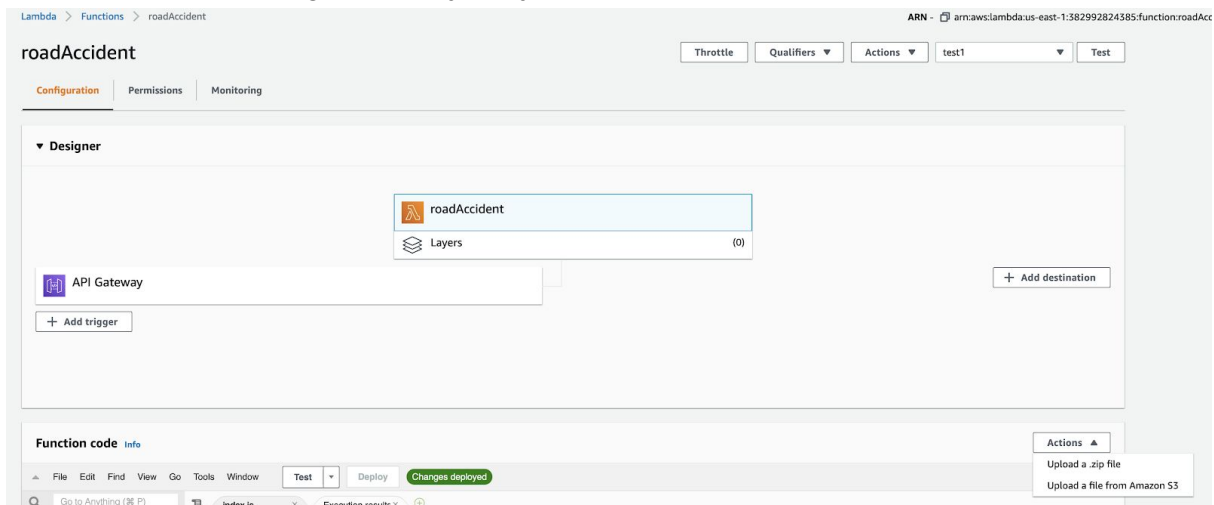
☒ Create a new role with basic Lambda permissions
 ☐ Use an existing role
 ☐ Create a new role from AWS policy templates

Role creation might take a few minutes. Please do not delete the role or edit the trust or permissions policies in this role.

Lambda will create an execution role named <myFunctionName>-role-njnt6js, with permission to upload logs to Amazon CloudWatch Logs.

- 6.3. Choose create from Aws policy template to choose policies from template
- 6.4. Then choose simple micro service permission
- 6.5. After the function is created click on the function we just created

- 6.6. On the Function code block click on top right and select upload zip file for uploading our Nodejs project zip file.



7. Prepare Nodejs Lambda project on local machine

- 7.1. Create normal nodejs project which contain package.json file and index.js file
- 7.2. Since we going to use some dependencies such as "Athena-express" we need to install dependencies on our Nodejs project using command this step require npm installed in your local machine
- ```
$ npm install athena-express --save
```
- 7.3. The node\_modules directory will be created after the command executed and the package. json file will update as follow

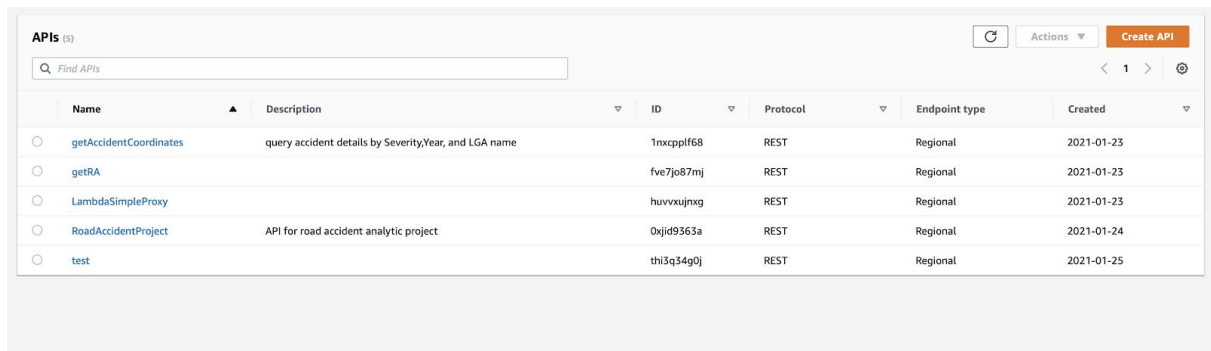
```
{
 "name": "lamdafunction",
 "version": "1.0.0",
 "description": "",
 "main": "index.js",
 "scripts": {
 "start": "node index.js"
 },
 "dependencies": {
 "athena-express": "^6.0.4"
 }
}
```



- 7.4. Add the index.js file path on tags "main" and add execute command on start tag in package.json file like in the image above.
- 7.5. Zip the project and upload on Aws console follow step 6.6.
- 7.6. After uploading the zip file click on deploy to deploy lambda function.

## 8. Create API Gateway

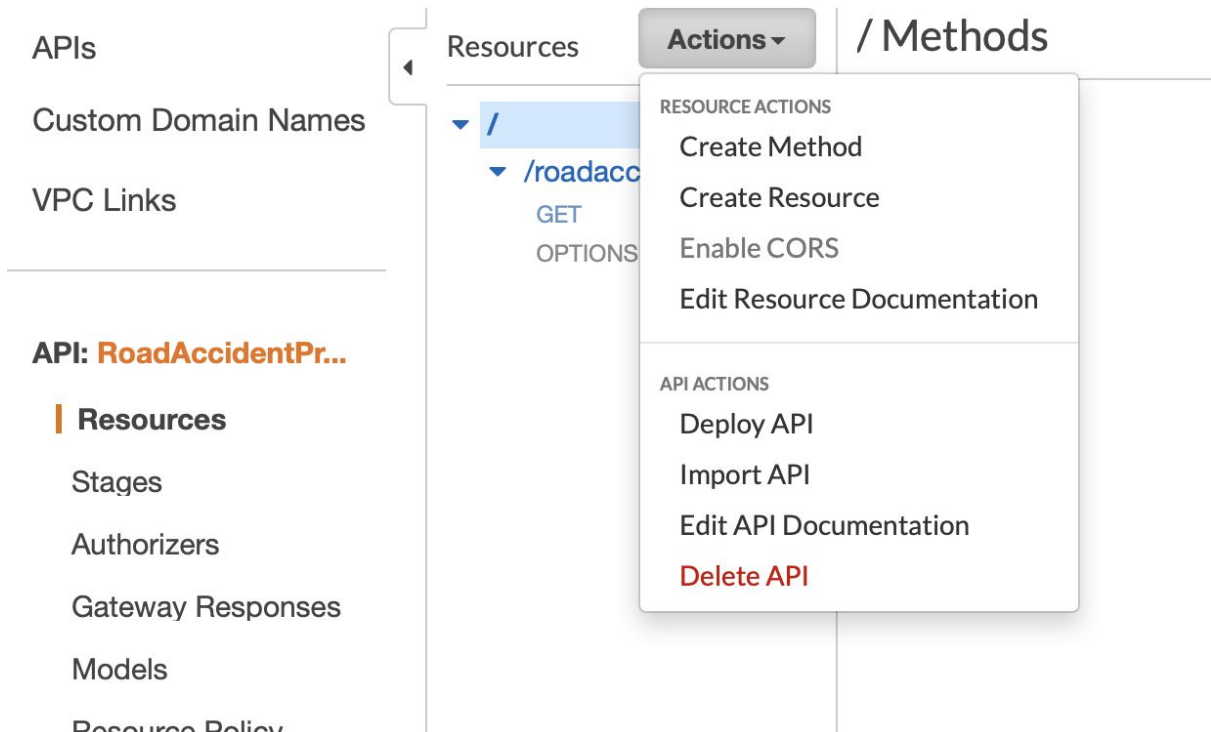
- 8.1. On Aws API Gateway console click on create API then choose Rest API (don't select private)
- 8.2. Then fill in the API name and save
- 8.3. Select the new created API from the list



The screenshot shows the AWS API Gateway console with a table of APIs. The table has columns for Name, Description, ID, Protocol, Endpoint type, and Created. There are five APIs listed: getAccidentCoordinates, getRA, LambdaSimpleProxy, RoadAccidentProject, and test.

| Name                   | Description                                           | ID         | Protocol | Endpoint type | Created    |
|------------------------|-------------------------------------------------------|------------|----------|---------------|------------|
| getAccidentCoordinates | query accident details by Severity,Year, and LGA name | 1nxcpllf68 | REST     | Regional      | 2021-01-23 |
| getRA                  |                                                       | fve7jo87mj | REST     | Regional      | 2021-01-23 |
| LambdaSimpleProxy      |                                                       | huvvxujnsg | REST     | Regional      | 2021-01-23 |
| RoadAccidentProject    | API for road accident analytic project                | 0xjd9363a  | REST     | Regional      | 2021-01-24 |
| test                   |                                                       | thi3q34g0j | REST     | Regional      | 2021-01-25 |

- 8.4. Click on Actions and select Create resource from the drop down list



- 8.5. Fill in resource name and check Enable API gateway CORS

### New Child Resource

Use this page to create a new child resource for your resource. 🚧

Configure as ☒ proxy resource

Resource Name\*

My Resource

Resource Path\*

/ my-resource

You can add path parameters using brackets. For example, the resource path {username} represents a path parameter called 'username'. Configuring {/proxy+} as a proxy resource catches all requests to its sub-resources. For example, it works for a GET request to /foo. To handle requests to /, add a new ANY method on the / resource.

- 8.6. Click on the new created resource and click on actions. This time select create Method.
- 8.7. Select GET and click on the check mark.
- 8.8. When the setup page appear check Lambda function , type in lambda function name that we created in step 6 also check select Use Lambda proxy integration them save

Choose the integration point for your new method.

Integration type ☒ Lambda Function ⓘ

☐ HTTP ⓘ

☐ Mock ⓘ

☐ AWS Service ⓘ

☐ VPC Link ⓘ

Use Lambda Proxy integration ☒ ⓘ

Lambda Region

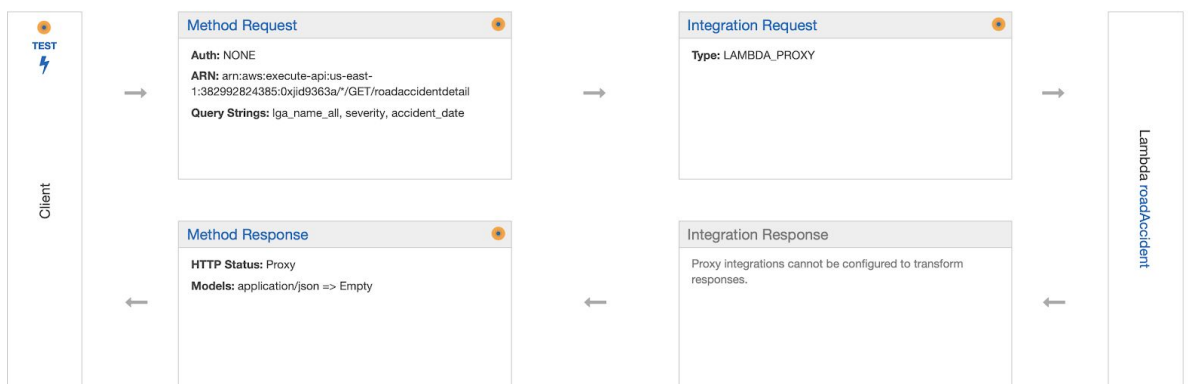
Lambda Function

Use Default Timeout ☒ ⓘ

[Save](#)

- 8.9. Select new created method on the list then click on Method Request

/roadaccidentdetail - GET - Method Execution



- 8.10. Create Query string parameters as follow

← Method Execution /roadaccidentdetail - GET - Method Request

Provide information about this method's authorization settings and the parameters it can receive.

Settings ⓘ

Authorization NONE ⓘ

Request Validator NONE ⓘ

API Key Required false ⓘ

▼ URL Query String Parameters ⓘ

| Name          | Required                 | Caching                  |  |
|---------------|--------------------------|--------------------------|--|
| accident_date | <input type="checkbox"/> | <input type="checkbox"/> |  |
| lga_name_all  | <input type="checkbox"/> | <input type="checkbox"/> |  |
| severity      | <input type="checkbox"/> | <input type="checkbox"/> |  |

[Add query string](#)

▶ HTTP Request Headers

▶ Request Body

▶ SDK Settings

- 8.11. Click Action again. Now select deploy API
- 8.12. Use Invoke URL to call our api which is already deployed

Invoke URL: <https://0xjld9363a.execute-api.us-east-1.amazonaws.com/roadAccidentAnalytic>

Settings

Logs/Tracing

Stage Variables

SDK Generation

Export

Deployment History

Documentation History

Canary

Cache Settings

Enable API cache

Default Method Throttling

Choose the default throttling level for the methods in this stage. Each method in this stage will respect these rate and burst settings. Your current account level throttling rate is **10000** requests per second with a burst of **5000** requests. [Read more about API Gateway throttling](#)

Enable throttling

Rate

10000

requests per second

Burst

5000

requests

Web Application Firewall (WAF) [Learn more.](#)

Select the Web ACL to be applied to this stage.

Web ACL

None

Create Web ACL

Client Certificate

Select the client certificate that API Gateway will use to call your integration endpoints in this stage.

Certificate

None

Save Changes

## 9. Install SSL certificate on Elastic Beanstalk Load Balancer using ACM and Route53


- 9.1. Valid ssl certificate is required inorder to display Aws Quicksight embed dashboard correctly
- 9.2. Request domain name and create hosted zone
  - 9.2.1. Registered domain using Amazon Route 53 waits until the domain has been created.
  - 9.2.2. Create a hosted zone by putting your domain name and select public hosted zone type.
  - 9.2.3. Go to AWS Certificate Manager and request a certificate.
  - 9.2.4. Add your domain name in step1
  - 9.2.5. In step 2,select DNS validation.
  - 9.2.6. For step 3,4 and 5, use everything as default.
  - 9.2.7. Click on the arrow in front of the Domain name. You will see the Name and Value of CNAME.

Certificates

AWS Certificate Manager logs domain names from your certificates into public certificate transparency (CT) logs when renewing certificates. You can opt out of CT logging. [Learn more](#)

[Request a certificate](#) [Import a certificate](#) [Actions](#)


« < Viewing certificates 1 to 4 > »

|                          | Name                                                                              | Domain name | Additional names | Status             | Type          | In use? | Renewal eligibility |
|--------------------------|-----------------------------------------------------------------------------------|-------------|------------------|--------------------|---------------|---------|---------------------|
| <input type="checkbox"/> |  | www.tet.com | -                | Pending validation | Amazon Issued | No      | Ineligible          |

Status

**Validation not complete**

The status of this certificate request is "Pending validation". Further action is needed to validate and approve the certificate. [Learn more](#).

| Domain                                                                                        | Validation status  |
|-----------------------------------------------------------------------------------------------|--------------------|
|  www.tet.com | Pending validation |

Add the following CNAME record to the DNS configuration for your domain. The procedure for adding CNAME records depends on your DNS service Provider. [Learn more](#).

| Name                                           | Type  | Value                                                           |
|------------------------------------------------|-------|-----------------------------------------------------------------|
| _d80c72d81ec501e8a3290b601fc3494a.www.tet.com. | CNAME | _1c0c6b82f30f4331659ca82cc72f615.vtqthvjcp.acm-validations.aws. |

**Note:** Changing the DNS configuration allows ACM to issue certificates for this domain name for as long as the DNS record exists. You can revoke permission at any time by removing the record. [Learn more](#).

- 9.2.8. Go to AWS Route 53 Hosted zones.
- 9.2.9. Click on the domain name you want to certify.
- 9.2.10. Select create record.
- 9.2.11. Paste Name and Value of CNAME into Record name and Value. In Record type select CNAME. Then, create record

Route 53 > Hosted zones > cloud-project-js-nr.com > Create record

**Quick create record** [Info](#) [Switch to wizard](#) [Add another record](#)

▼ Record 1 [Delete](#)

Routing policy [Info](#)  
Simple routing ▼

Record name [Info](#)  
 .cloud-project-js-nr.com  
Valid characters: a-z, 0-9, ! " # \$ % & ' ( ) \* + , - / : ; < = > ? @ [ \ ] ^ \_ ` { | } . ~

Alias [Info](#)  
☐

Record type [Info](#)  
CNAME – Routes traffic to another domain n... ▼

Value [Info](#)  
  
Enter multiple values on separate lines.

TTL (seconds) [Info](#)  
  
1m 1h 1d  
Recommended values: 60 to 172800 (two days)

[Cancel](#) [Create records](#)

- 9.3. Allow QuickSight to be an embedded dashboard in your domain.
  - 9.3.1. When your domain gets certificates go to quicksight.
  - 9.3.2. Click on your username on top right corner and select Manage QuickSight.
  - 9.3.3. Select Domain and Embedding and add your domain that you want to embed the dashboard and check on Include subdomains.

Account name: jakrapun  
Edition: Enterprise

Manage users  
Your subscriptions  
SPICE capacity  
Account settings  
Security & permissions  
Manage VPC connections  
Mobile settings

[Domains and Embedding](#)

#### Manage domains for embedded dashboards

Embedded dashboards only work if they originate from domains you explicitly allow.

##### Domain

☒ Required

☐ Include subdomains ⓘ

Add

#### 9.4. Request Certificate

#### 9.5. Configure Elastic Beanstalk Load Balancer

9.5.1. On Elastic Beanstalk console click on the project environment

9.5.2. Select configuration from the list

9.5.3. Click on edit button on the Load balancer block

9.5.4. On listener block click on add listener

Elastic Beanstalk > Environments > node-express-dev > Configuration

### Modify Application Load Balancer

**Listeners**  
You can specify listeners for your load balancer. Each listener routes incoming client traffic on a specified port using a specified protocol to your environment processes. By default, we've configured your load balancer with a standard web server on port 80.

Actions ▼

+ Add listener

| <input type="checkbox"/> | Port | Protocol | SSL certificate        | Default process | Enabled                             |
|--------------------------|------|----------|------------------------|-----------------|-------------------------------------|
| <input type="checkbox"/> | 443  | HTTPS    | elastic-beanstalk-x509 | default         | <input checked="" type="checkbox"/> |
| <input type="checkbox"/> | 80   | HTTP     | --                     | default         | <input checked="" type="checkbox"/> |

9.5.5. In put port number 443 , select https protocol and select ssl we created from step 9.3

9.5.6. The ssl policies that Aws recommended by Aws is ELBSecurityPolicy-2016-08 so we choose that one.

## Application Load Balancer listener

Port

443


Listener port must be unique.

Protocol

The transport protocol that the load balancer uses for routing incoming traffic from clients.

HTTPS

SSL certificate

www.cloud-project-js-nr.com - 25bc0374-8bea-4e... 

SSL policy

The Secure Sockets Layer (SSL) negotiation configuration, known as a security policy, that this load balancer uses to negotiate SSL connections with clients.

ELBSecurityPolicy-2016-08

Default process

The process to which the listener routes traffic by default, when the message path doesn't match any custom listener rule.

default

Cancel

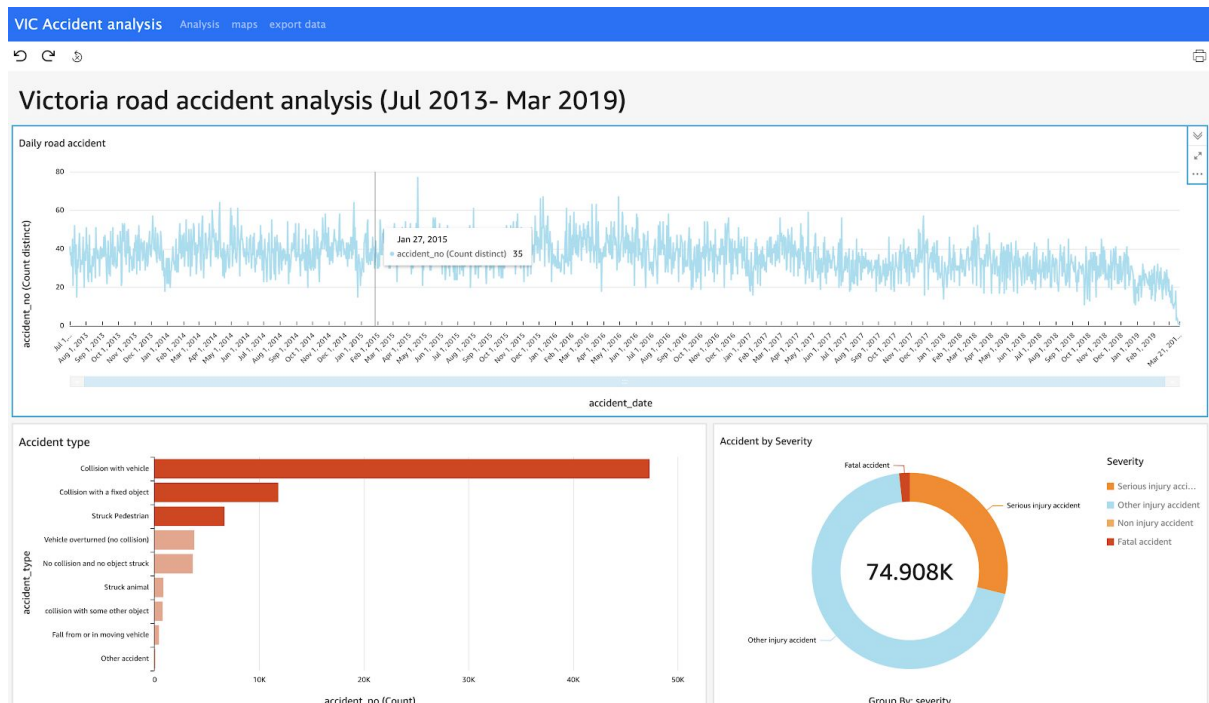
Add

- 9.5.7. After click add , click apply at the end of the page. Now we can access our web application by https and Aws Quicksight embed url display correctly.

# User manual

## Road Accident Analysis

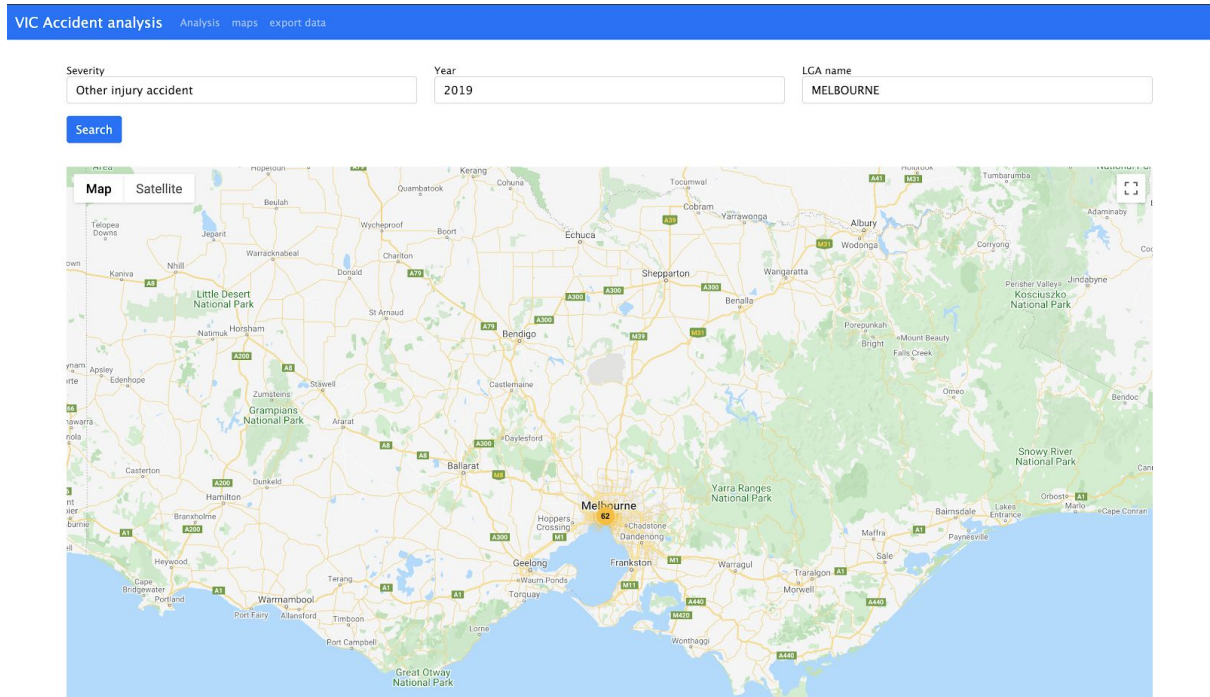
1. The main page will display data analysis of past 5 years road accident of Victoria



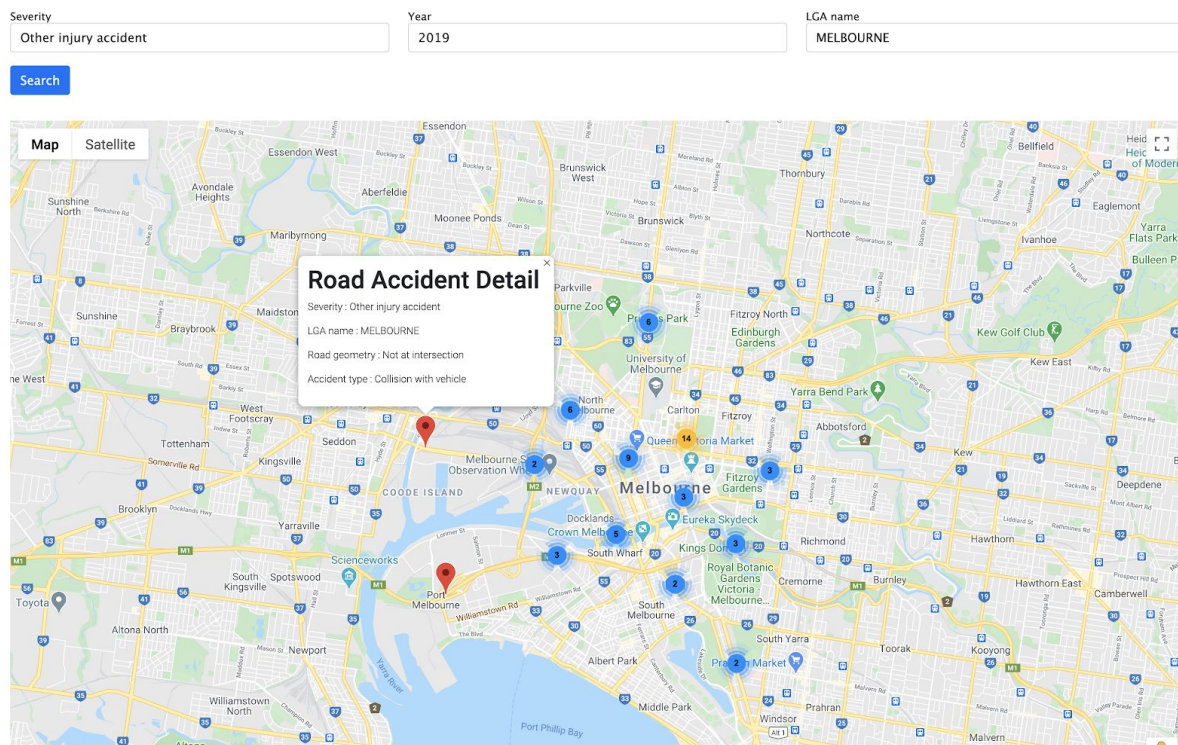


## Interactive Map

1. Select map form the top navigation bar
2. The web will display interactive map as the images displayed below



3. Select Severity, year and LGA name from dropdown list to choose to display location on the map.
4. The number show cluster of the accident location
5. Click on cluster to expand more location
6. Click on the pinned to see more detail of the accident





## Export Report

1. Select export data from top navigation bar
2. The page will display table of example 200 records
3. Click on export csv link at the top of table

VIC Accident analysis Analysis maps export data

Example exported data set 200 records

[Export CSV](#) Search:

| Accident No. | Severity                | LGA            | Road Geometry         | Accident Type                     | Accident Date |
|--------------|-------------------------|----------------|-----------------------|-----------------------------------|---------------|
| T20130013608 | Serious injury accident | KNOX           | Multiple intersection | Collision with vehicle            | 1/7/2013      |
| T20130013611 | Other injury accident   | GEE LONG       | T intersection        | Struck Pedestrian                 | 1/7/2013      |
| T20130013617 | Serious injury accident | CASEY          | T intersection        | Collision with vehicle            | 1/7/2013      |
| T20130013620 | Serious injury accident | BAW BAW        | Not at intersection   | Collision with a fixed object     | 1/7/2013      |
| T20130013621 | Other injury accident   | BENDIGO        | Not at intersection   | Struck animal                     | 1/7/2013      |
| T20130013622 | Other injury accident   | FRANKSTON      | T intersection        | Collision with vehicle            | 1/7/2013      |
| T20130013630 | Other injury accident   | MITCHELL       | Not at intersection   | Vehicle overturned (no collision) | 1/7/2013      |
| T20130013638 | Serious injury accident | EAST GIPPSLAND | Not at intersection   | Collision with a fixed object     | 1/7/2013      |
| T20130013641 | Fatal accident          | GEE LONG       | T intersection        | Collision with a fixed object     | 1/7/2013      |
| T20130013644 | Fatal accident          | BAYSIDE        | Cross intersection    | Struck Pedestrian                 | 1/7/2013      |

Previous 1 2 3 4 5 ... 20 Next

## Reference.

[1] *Crashes Last Five Years*, Victorian Department of Transport Open Data, 2019.[Online] Available:

[https://vicroadsopendata-vicroadsmaps.opendata.arcgis.com/datasets/c2a69622ebad42e7baaa8167daa72127\\_0/data](https://vicroadsopendata-vicroadsmaps.opendata.arcgis.com/datasets/c2a69622ebad42e7baaa8167daa72127_0/data)

[2] Spatial Vision, "Victorian Crash Statistics", [spatialvision.com.au](http://spatialvision.com.au).

<https://www.spatialvision.com.au/crashstats/> (accessed Jan. 30, 2021).

[3] ArcGIS, "My Map", [arcgis.com](http://arcgis.com)

[https://www.arcgis.com/home/webmap/viewer.html?panel=gallery&suggestField=true&url=https%3A%2F%2Fservices2.arcgis.com%2F18ajPSI0b3ppsmMt%2Farcgis%2Frest%2Fservices%2FCrashes\\_Last\\_Five\\_Years%2FFeatureServer%2F0](https://www.arcgis.com/home/webmap/viewer.html?panel=gallery&suggestField=true&url=https%3A%2F%2Fservices2.arcgis.com%2F18ajPSI0b3ppsmMt%2Farcgis%2Frest%2Fservices%2FCrashes_Last_Five_Years%2FFeatureServer%2F0) (accessed Jan. 30, 2021).